REMARKS

Claims 1-8, 10-15, 17-23, and 27 are currently pending in the subject application and are presently under consideration. Claims 1, 6, 10, 12, 14-15, 17, 19, 23, and 27 have been amended as shown on pp. 2-8 of the Reply. Favorable reconsideration of the subject patent application is respectfully requested in view of the comments and amendments herein.

I. Rejection of Claim 27 Under 35 U.S.C. §101

Claim 27 stands rejected under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject matter. Withdrawal of this rejection is respectfully requested for at least the following reasons. Independent claim 27 has been amended herein to recite at least one computer readable storage medium that stores computer executable instructions, which are executed by at least one processor. It is submitted that such amendments clearly place these independent claim 27 within the bounds of statutory subject matter in accordance with 35 U.S.C. § 101; and this rejection should be withdrawn.

II. Rejection of Claims 1-8 and 9-14 Under 35 U.S.C. §103(a)

Claims 1-8 and 9-14 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Arnold, *et al.* (US 6,393,497), in view of Hollander, *et al.* (US 6,823,460), in further view of Clarke, *et al.* (US 2002/0035642). Withdrawal of this rejection is respectfully requested for at least the following reasons. Arnold, *et al*. alone or in combination with Hollander, *et al.* and/or Clarke, *et al.* does not teach or suggest all aspects of the subject claims.

> ***[T]he prior art reference (or references when combined) must teach or suggest all the claim limitations.*** *See* MPEP § 706.02(j). *See also KSR Int'l Co. v. Teleflex, Inc*., 550 U. S. ____, 04-1350, slip op. at 14 (2007). The teaching or suggestion to make the claimed combination and the reasonable expectation of success must be found in the prior art and not based on applicant's disclosure. *See In re Vaeck,* 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991) (emphasis added).

Applicant's subject claims relate to a system and method that provide application developer coded extensible proxies that have access to method interception and remoting functionality. In particular, the system facilitates application developers building proxies by

providing one or more base classes that have access to intercepting and/or remoting functionality from which application developers can inherit. The base class proxy object can provide system level functions like remoting infrastructure functionality (*e.g.,* message passing, sinks, message synchronization) and intercepting infrastructure functionality (*e.g.,* call stack access, acquisition of process control) to an application developer. The base class proxy provides an interface to system level functions like call intercepting by providing one or more methods that can be overridden and which will invoked when the method call to be intercepted is invoked on the object imaged by the application code generic proxy. The interface facilitates not only receiving control when the method to be intercepted is invoked, but also facilitates receiving information concerning the method call. To this end independent claim 1, as amended, recites ***the application code generic proxy performs proxy pre-processing that includes custom, user-directed application processing to at least one of monitor or control the processing of a message between the application code generic proxy and the object that includes controlling at least one amount of data sent to the object, a type of data sent to the object or which objects are invoked, in a manner to reduce response time.*** Further, independent claim 1 recites ***a method call interceptor that intercepts a method call to an object and that directly routes the method call to an application code generic proxy that is an image of a local object, and the application code generic proxy that receives an intercepted method call, invokes a method on the object without crossing a remoting boundary.*** Similarly, independent claim 14 recites ***a method call intercepting component that intercepts a method call to an object and directly routes the method call to an application code generic proxy that is an image of a local object, the application code generic proxy component that receives an intercepted method call, invokes a method on the object without traversing a remoting boundary, and the application code generic proxy component functionality modified by the application code based at least in part on the intercepted method call to perform custom user-directed pre-processing that controls the processing of a message between the application code generic proxy and the object, which includes achieving a reduced response time by controlling at least one of amount of data sent to the object, a type of data sent to the object or which objects are invoked.*** The cited art, alone or in combination, fails to teach or suggest these novel aspects.

Arnold, *et al.* relates to utilization of a smart proxy as a wrapper around a stub in a distributed system wherein, a caller receives a smart proxy including a stub as an embedded

object instead of receiving only the stub as a result of a remote procedure call. Specifically, the smart proxy performs predefined processing associated with a remote procedure call, the processing possibly occurring before, during, or after a response to the call. However, Arnold, *et al.* does not disclose or suggest intercepting a method call to an object and directly routing the method call to a generic proxy that invokes a method on the object without crossing a remoting boundary. In contrast, as indicated in col. 10, lines 37-42, the Arnold, *et al.* states that a client machine transmits a call or request for a particular object to a server, which in turn responds by returning a smart proxy with an embedded stub, the returned smart proxy can thereafter act as a representation of the requested object. Thus, until the server responds with the smart proxy, no proxy ever exists. In contrast, the claimed subject matter provides that a method call to an object is intercepted and directly routed to an extant generic proxy wherein the generic proxy can invoke a method on the object. It is submitted that this is a discernable and non-obvious distinction between the primary document and the claimed subject matter. In addition, Arnold, *et al.* relates to a smart proxy that can be downloaded onto a client machine and can be utilized by the client machine to determine if preprocessing is required. If required, the processing is performed locally by the client machine using the smart proxy. (*See* column 10, lines 43-46.) However, Arnold, *et al.* fails to disclose custom user-directed pre-processing that controls the processing of a message between the application code generic proxy and the object, which includes achieving a reduced response time by controlling at least one of amount of data sent to the object, a type of data sent to the object or which objects are invoked, as recited in the subject claims. Further, Hollander, *et al.* and/or Clarke, *et al.* do not cure these deficiencies, as explained below.

Hollander, *et al.* merely relates to a method of intercepting application program interface, including dynamic installation of associated software, within the user portion of an operating system. Specifically, API functions called by user applications are not allowed to execute unless the calling process has the requisite authority and privilege. Moreover, API function arguments are processed and decision is made by a Pre-Entry routine whether the API function is allowed to execute in the present environment. However, Hollander, *et al.* is silent with respect to a method call interceptor that intercepts a method call to an object and directly routes the method call to an application code generic proxy, which is an image of a local object. Further, Hollander, *et al.* fails to teach or suggest an application code generic proxy that receives an intercepted method

call and invokes a method on the object without crossing a remoting boundary. Additionally, the system and method disclosed by Hollander, *et al.* does not teach or suggest processing of a message between the application code generic proxy and the object that includes controlling at least one amount of data sent to the object, a type of data sent to the object or which objects are invoked, in a manner to reduce response time, as recited in the subject claims. The system disclosed by Hollander, *et al.* simply determines whether the API function is allowed to execute in the present environment based on the type and value of the arguments passed by the calling applications but fails to teach or suggest an application code generic proxy that performs proxy pre-processing comprising transaction processing and machine learning to determine an amount of data sent to the object, determine a type of data sent to the object and/or determine which objects are invoked before invoking a method on the object. Furthermore, the system disclosed by Hollander, *et al.* relates to identification of the type and value of the arguments passed by the calling applications to determine if the API function is allowed to execute in the present environment but does not teach or suggest controlling type or value of the arguments, such that, a determined type/value is passed.

Clarke, *et al.* relates to a system and method that manages network traffic by employing an intermediary node on a data communications network, such as a proxy, that may implement a flow control algorithm to control network congestion. In particular, the intermediary node receives messages destined for servers or other upstream nodes from clients or other downstream nodes and determines whether to forward the messages based on the flow control algorithm, which may be adaptive based on network performance and/or usage. Further, the system includes a server that returns an error response when it receives a request that it cannot handle from the client. This response is passed back to the client *via* a proxy, which recognizes the response type and learns a back off time for the server. Incase a disparate client sends a request to the server, the proxy returns a back off signal to the disparate client and reduces the number of requests reaching the congested server. However, Clarke, *et al.* fails to disclose direct routing of a method call to an application code generic proxy, which is an image of a local object and does not teach or suggest an application code generic proxy that invokes a method on the object without crossing a remoting boundary. Further, Clarke, *et al.* is silent with respect to controlling at least one amount of data sent to the object, a type of data sent to the object or which objects are invoked, in a manner to reduce response time, as recited in the subject claims and thus fails to

remedy the aforementioned deficiencies of Arnold, *et al.* and/or Hollander, *et al.* with respect to independent claims 1 and 14.

Applicants' subject claims, in contrast to the cited art, relate to a system that employs application developer coded extensible proxies that have access to method interception and remote functionality and data. The disclosed system provides a proxy that includes a customized proxy component that can be written in application code by application programmers and can have access to the interception and routing functionality provided by system code. The customized proxy component can be, therefore, operable to adapt and/or extend the functionality provided by a conventional proxy. Specifically, application code may be employed in actions including, but not limited to, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation. (*See* page 11, lines 3-25.) Since the customized proxy had the opportunity to perform proxy pre-processing, ***the invocation includes optimization of the amount of data sent to the remote object, the type of data sent to the remote object and which remote object(s), if any, are invoked***. For example, in the web browser/stock market data feed example, a user may have previously inquired about a stock price for a company named AABBCC company. Before the stock price could be retrieved, the remote object may have had to look up a ticker symbol for the company (*e.g.,* AACo), and that ticker symbol may have been cached in the customized proxy. Thus, the ticker symbol, rather than the name of the company, may be sent to the remote object, which can eliminate a duplicate lookup in the remote object. By way of further illustration, proxy pre-processing performed by the customized proxy can have determined that a second remote object may be better suited to handle the method invoked by the method caller than a first remote object, and thus the customized proxy may invoke the method on the second remote object instead of the first remote object. For example, returning to the browser/stock market data feed example, the customized proxy may have determined that recent stock market inquiries on a first remote object have taken sixty seconds, while a response time of five seconds is desired. Thus, the customized proxy may acquire the stock market price from a different remote object and compare the average response times to determine how more optimal response times can be achieved. Thus, increases in performance may be achieved for the method caller. The cited art, alone or in combination, is silent with respect to these novel aspects.

In addition, dependent claim 12 recites ***the proxy post-processing comprises at least one of auditing, transaction processing, object migration, object persisting, monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations or machine learning involved in optimizing remote method call invocation***. Moreover, where a result is passed to the method caller, the customized proxy disclosed by the subject system can perform processing and pass the result to the proxy. For example, the customized proxy can receive data in a first format (*e.g.,* day, month, year) from the remote object while the method caller may be expecting data in a second format (*e.g.,* month, day, year), and thus the customized proxy can translate between the formats before passing the result to the proxy. Thus, the method caller may benefit by gaining access to a remote object and by receiving data in an expected format that removes the need to reformat such data at the method caller. Proxy post-processing can include, but is not limited to monitoring remote method calls, caching local data, caching remote data, controlling remote method call invocations and machine learning involved in optimizing remote method call invocation. (*See* page 13, lines 8-22.) The cited art is silent with respect to this novel aspect.

In view of at least the foregoing, it is readily apparent that Arnold, *et al*., Hollander, *et al.* and/or Clarke, *et al*., alone or in combination, do not teach or suggest applicants' claimed subject matter as recited in independent claims 1 and 14 (and associated dependent) and thus fail to make obvious the subject claims. Accordingly, withdrawal of this rejection is respectfully requested.

III. Rejection of Claims 15 and 20-22 Under 35 U.S.C. §103(a)

Claims 15 and 20-22 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Colyer (US 5,903,725), in view of Clarke, *et al.* (US 2002/0035642). Withdrawal of this rejection is respectfully requested in view of at least the following reasons. The cited art fail to teach or suggest each and every feature of the subject claims. Specifically, independent claim 15 recites ***routing the method call to the application code generic proxy without traversing a remoting boundary, wherein the application code generic proxy is an image of a local object; adapting the application code generic proxy functionality based at least in part on the method call, the application code generic proxy performs custom user-directed proxy pre-processing comprising transaction processing and machine learning to control at least one of an amount***

of data sent to the object, a type of data sent to the object or which objects are invoked; and, invoking the method on the object based in part on the pre-processing. As discussed *supra*, Clarke, *et al.* and Hollander, *et al.*, fail to teach or suggest these novel aspects. Further, Colyer relates to mechanisms for protecting a server against invalid usage of proxy objects after malfunction of a server and for transparently re-creating proxy objects in a client of a client-server distributed processing system. However, Colyer fails to disclose interception of a method call that is made accessible to a developer to at least one of adapt or extend functionalities of a proxy and does not teach or suggest controlling an amount of data sent to the object, a type of data sent to the object or which objects are to be invoked, as recited in claim 15. Furthermore, Colyer is silent with respect to routing the method call to the application code generic proxy without traversing a remoting boundary, wherein the application code generic proxy is an image of a local object and thus fails to make up for the aforementioned deficiencies of Clarke, *et al.* and Hollander, *et al.* Accordingly, withdrawal of this rejection is requested.

IV. Rejection of Claims 17-19, 23, and 27 Under 35 U.S.C. §103(a)

Claims 17-19, 23, and 27 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Colyer (US 5,903,725), Clarke, *et al.* (US 2002/0035642) and Hollander, *et al.* (US 6,823,460), in further view of Arnold, *et al.* (US 6,393,497). Withdrawal of this rejection is respectfully requested in view of at least the following reasons. The cited references, alone or in combination, fail to teach or suggest each and every limitation of applicants' claimed invention. Independent claim 23 recites ***employing the application code generic proxy to perform custom user-directed proxy pre-processing comprising at least machine learning for optimization of remote method call invocation prior to invoking a method on a object such that response time is reduced, wherein the optimization of the remote method call includes controlling at least one of amount of data sent to the object, type of data sent to the object or which objects are invoked; routing the method call to the application code generic proxy that is an image of a local object, without traversing a remoting boundary; and, invoking the method on the object based in part on the custom user-directed proxy pre-processing.*** Similarly, independent claim 27 recites ***means for the application code generic proxy that is an image of a local object, to receive the intercepted method call without traversing a remoting boundary, and, means for the application code generic proxy to perform proxy pre-processing that includes load-***

balancing, transaction processing and machine learning to reduce response time prior to utilizing the means for the application code generic proxy to invoke the method on the object, wherein the proxy pre-processing is user-directed and customized for controlling at least one of amount of data sent to the object, type of data sent to the object or which objects are invoked. Additionally, as discussed *supra,* independent claim 15 recites ***routing the method call to the application code generic proxy without traversing a remoting boundary, wherein the application code generic proxy is an image of a local object; adapting the application code generic proxy functionality based at least in part on the method call, the application code generic proxy performs custom user-directed proxy pre-processing comprising transaction processing and machine learning to control at least one of an amount of data sent to the object, a type of data sent to the object or which objects are invoked; and, invoking the method on the object based in part on the pre-processing.*** As discussed above, Colyer and Clarke, *et al.*, alone or in combination do not disclose a system or method that intercepts a method call and controls at least one of, an amount of data sent to the object, a type of data sent to the object, or objects that are to be invoked to respond to the method call. Furthermore, Arnold, *et al.* and Hollander, *et al.* fail to remedy the aforementioned deficiencies of Colyer and Clarke, *et al*. Accordingly, it is respectfully requested that this rejection be withdrawn.

CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments and amendments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP243US].

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

TUROCY & WATSON, LLP

/Thomas E. Watson/
Thomas E. Watson
Reg. No. 43,243

TUROCY & WATSON, LLP
127 Public Square
57TH Floor, Key Tower
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731